

# Internet Engineering Task Force (IETF)での ハッカソンと Secure Update の 技術開発の紹介

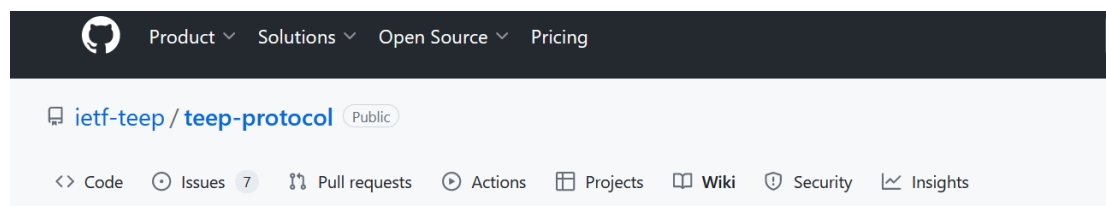
2023年5月

# 直近の活動の紹介

- Internet Engineering Task Force (IETF) で 2 つの活動
  - 1) IETF で標準化ドラフト(Internet-Drafts, I-Ds)のための技術開発と実装
  - 2) I-Ds の著者としてドラフトが正式な標準化文章(RFC)になるよう継続的なアップデート、RFC化はもうすぐ

# IETF での技術開発、TEEPの紹介 (1/3)

- Trusted Execution Environment Provisioning (TEEP) Protocol を開発
- <https://github.com/ietf-teep/teep-protocol/wiki>



出るまでが長かった  
3年ほど遅れた  
辛かった

## Home

Akira Tsukamoto edited this page 3 hours ago · 23 revisions

## IETF TEEP Protocol Wiki Page

### Introduction

To simplify the life of developers interacting with Trusted Applications in a Trusted Execution Environment (TEE), an interoperable protocol for managing TAs running in different TEEs of various devices is needed - the TEEP protocol.

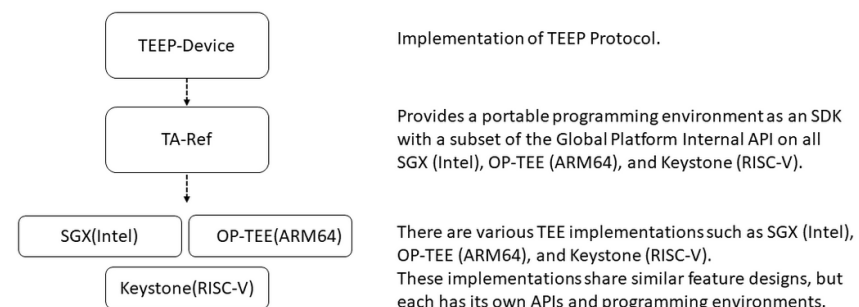
[Introduction, objective and use cases of TEEP](#)

The charter page of the TEEP working group can be found at: <https://datatracker.ietf.org/wg/teep/about/>

These are the main working group documents:

- TEEP Architecture: <https://datatracker.ietf.org/doc/draft-ietf-teep-architecture/>
- TEEP over HTTP: <https://datatracker.ietf.org/doc/draft-ietf-teep-otrp-over-http/>
- TEEP Protocol: <https://datatracker.ietf.org/doc/draft-ietf-teep-protocol/>

## Relationship of TA-Ref and TEEP-Device

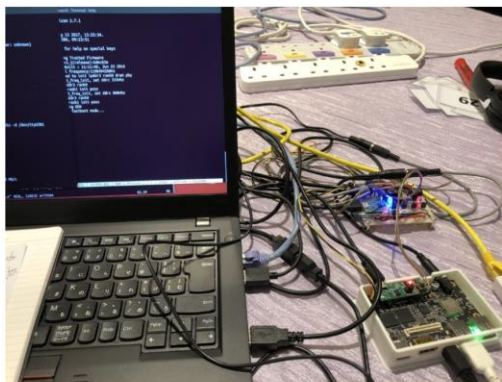


# IETF での技術開発、TEEPの紹介(2/3)

## On the Table

- IETF ハッカソンにて。セコムさんに大感謝。

### TEEP device



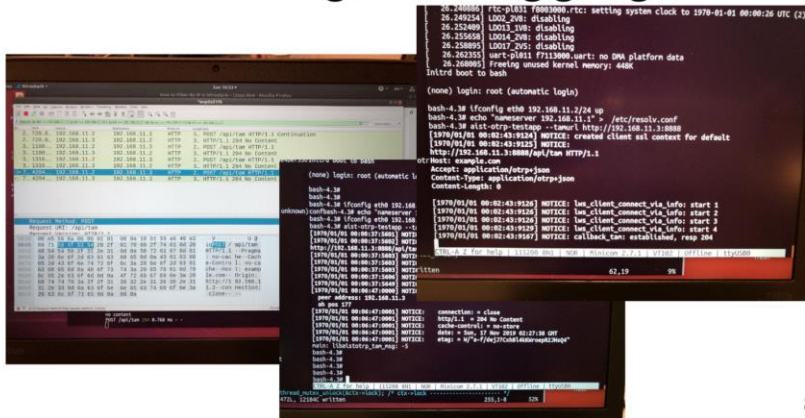
IETF 106 Hackathon - TEEP

ケーブル  
がたくさん



IETF 106 Hackathon - TEEP

### Hacking, Debugging!



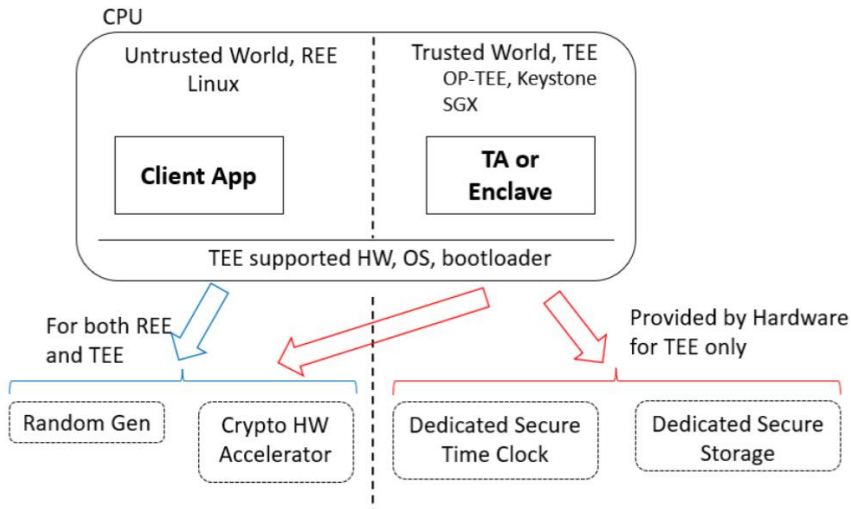
# IETF での技術開発、TEEPの紹介(3/3)

<https://github.com/ietf-teep/teep-protocol/wiki/files/ta-ref-open/ta-ref.pdf>

<https://github.com/ietf-teep/teep-protocol/wiki/files/teep-device-open/teep-device.pdf>

2

## 1.1.1 Assumption of hardware features of TA-Ref on TEE



The Secure Time Clock is the date and time clock hardware peripheral which updates monotonically provided separately from regular clock peripheral so the user application and OS on REE could not change the date and/or time. Many certificates of CA, license keys of purchased serial code, hardware enablement keys such as increasing the battery size of the electric cars are bound to the date. The easiest way for end users or attackers phishing the CAs and web sites, using the software and enabling the optional hardware feature without the payment is to change the value of the clock. The concrete date and time is especially important for the telemetry data.

1 Overview of TEEP-Device	1
1.1 Use Cases of TEEP	2
1.2 Features of TEEP-Device	3
1.3 Components of TEEP-Device and TA-Ref	3
1.3.1 TEEP-Device and TA-Ref Components on Keystone	4
1.3.2 TEEP-Device and TA-Ref Components on OP-TEE	4
1.3.3 TEEP-Device and TA-Ref Components on SGX	5
1.4 The design of TEEP Agent and sample TA of HELLO-TEEP-TA	5
2 Operation of TAM and device	6
3 Concise Binary Object Representation (CBOR) in TEEP-Device	7
3.1 Three format representations in TEEP and SUIT	7
3.2 TEEP Message format	8
3.3 SUIT Manifest format	9
4 Directory structure of source files	10
5 Consideration of build machine and development environment	11
5.1 How to select the build machine	11
5.2 How to setup an efficient development environment	12
6 Build TEEP-Device with Docker	13
6.1 Preparation for Docker	13
6.1.1 Install Docker	13



# IETF の Internet-Draft を日々更新

<https://datatracker.ietf.org/doc/draft-ietf-teep-protocol/>

<https://github.com/ietf-teep/teep-protocol>

draft-ietf-teep-protocol-13

TEEP  
Internet-Draft  
Intended status: Standards Track  
Expires: 2 November 2023

H. Tschofenig  
M. Pei  
Broadcom  
D. Wheeler  
Amazon  
D. Thaler  
Microsoft  
A. Tsukamoto  
1 May 2023

Trusted Execution Environment Provisioning (TEEP) Protocol  
draft-ietf-teep-protocol-13

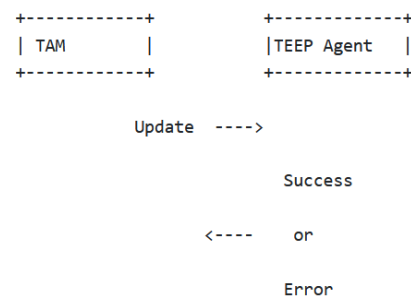
## Abstract

This document specifies a protocol that installs, updates, and deletes Trusted Components in a device with a Trusted Execution Environment (TEE). This specification defines an interoperable protocol for managing the lifecycle of Trusted Components.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-



## 4. Detailed Messages Specification

TEEP messages are protected by the COSE\_Sign1 or COSE\_Sign structure as described in Section 8.1. The TEEP protocol messages are described in CDDL format [RFC8610] below.

teep-message = \$teep-message-type .within teep-message-framework

```
teep-message-framework = [
  type: $teep-type / $teep-type-extension,
  options: { * teep-option },
  * any; further elements, e.g., for data-item-requested
]
```

teep-option = (uint => any)

; messages defined below:  
\$teep-message-type /= query-request  
\$teep-message-type /= query-response  
\$teep-message-type /= update  
\$teep-message-type /= teep-success  
\$teep-message-type /= teep-error

# TEEP とは (1/2)

## Acronyms

Trusted Execution Environment Provisioning (TEEP)

Software Updates for Internet of Things (SUIT)

Remote ATtestation ProcedureS (RATS)

- Target Audience

Vendors who develop products with CPU or SoC require Secure Update of software and data

- Lifecycle Management for Trusted Applications (Software) and Personalization data (Data)

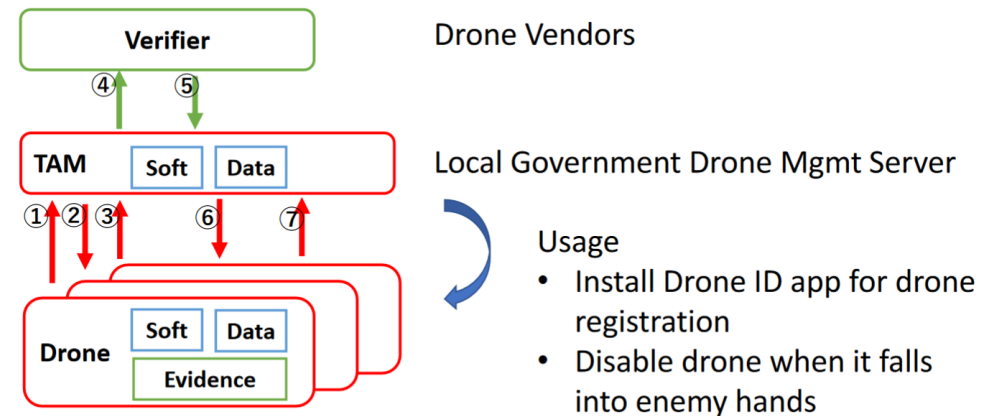
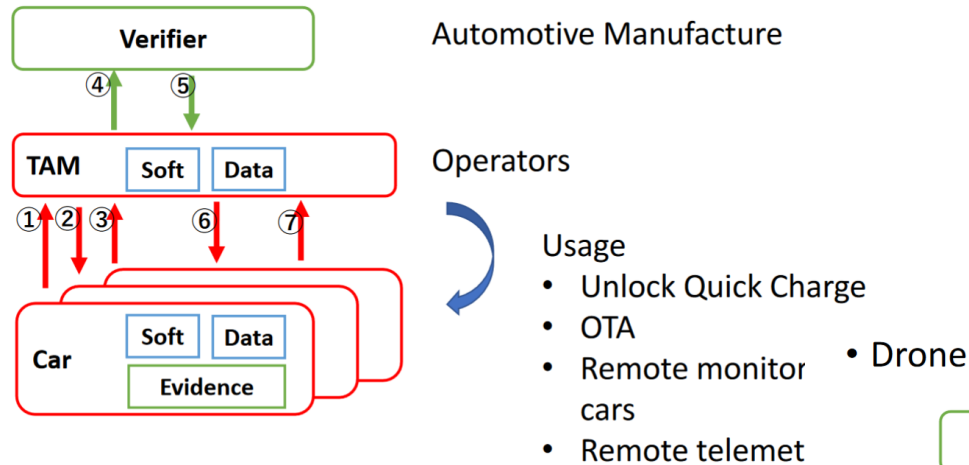
- Main objective is to manage Software and data in IoT devices to have latest version

- Before updates of the Software and Personalization data in IoT , the server check the trustworthiness of the IoT devices remotely whether it is compromised or not

# TEEP とは (1/2)

- <https://github.com/ietf-teep/teep-protocol/wiki/files/2023-IETF-TEEP-activity-2023-04-18-1.pdf>

- Automotive

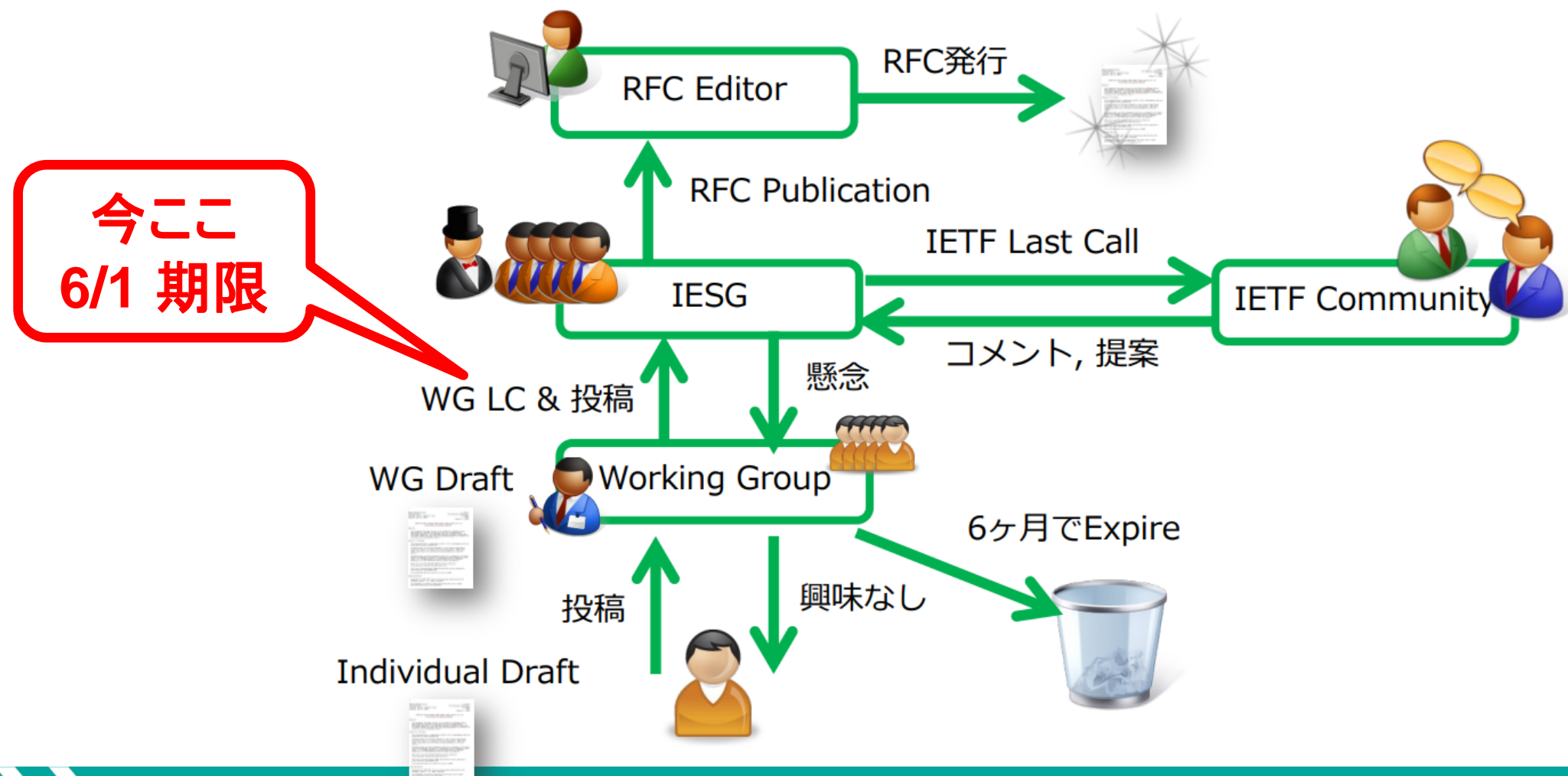




# I.D が RFC になるまでの流れ (1/2)

## RFC発行までのプロセス (1/2)

GMO CYBER SECURITY IERA E



# I.D が RFC になるまでの流れ (2/2)

## RFC発行までのプロセス (2/2)

GMO CYBERSECURITY IERAE

- **Individual Draft**

- Internet Draft (I-D) を執筆しIETF に投稿したら、まずこのステータス
- 適切と思われるWGで意見を求める

- **Working Group Draft**

- WGのCharterに合致かつWG memberからのコンセンサスを得るとWG Draft (WG item) に昇格
- その後、詳細内容についてMLなどで議論を行い、I-Dのブラッシュアップ
- WG Last Call を行いフィードバックを反映してIESGへ提出

- **IESG Process**

- 担当AD による Review 後、IETF Last Call を行い IETF Community 全体からのフィードバックを得る。その後、IESG Member による Review が実施される

- **RFC Editor Process**

- RFC発行に向けて、フォーマット確認。文章見直しなどの編集作業が実施される

# IETF での活動の内容

- 基本的な活動内容は 2002年に見た IETF の活動内容をまねた
- まねた内容は次ページ以降に
- IETF 開催毎に実装やハッカソンで見つかる課題を標準化ドラフト (Internet-Draft)に反映
- 実装はソフトウェアサプライチェーンを考慮
  - 1) Docker による開発環境の提供
  - 2) 全ソースにコピーライト、SPDX 情報、ライセンス条項を記述
  - 3) Makefile ならびに詳細なビルドドキュメント
  - 4) Git の全 commit log 提供
  - 5) CI 用スクリプト提供 (gitlab を使っていた)

# 大学で IETF 準拠テレビ会議システムのプロジェクトに参加

金が必要だった

今の zoom, teams と同じ機能

2001年に

音声、動画、テキストチャット、  
画面共有機能



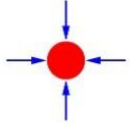
## CINEMA

### Columbia InterNet Extensible Multimedia Architecture

CINEMA is a set of SIP-based Internet multimedia servers for enterprise Internet telephony and multimedia system, consisting



sipd SIP proxy, redirect and registrar server



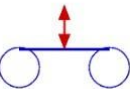
sipconf SIP multimedia conferencing server



sipum SIP voicemail/unified messaging server



sip323 SIP-H.323 protocol translator



rtspd RTSP media server

Latest version is cinema-1.21 **NEW**, released on April 05, 2002.

- For licensing information, please contact <http://www.sipcomm.com>.
- Installation instructions for Unix and Windows platforms.
- Frequently asked questions and trouble shooting.
- Documentation for sipd, sipconf, sipum, sip323, rtspd, sipua, and sipstone **NEW**.
- A complete technical report on CINEMA that describes the overall system architecture.
- System requirements:
  - For Windows: We recommend Windows 2000, but the software also works with Windows NT 4.0
  - Unix: any standard Solaris (5.8), Linux (e.g., RedHat 6.2, with RedHat 7.1 or later recommended), FreeBSD (4.3)
  - 64 MB or above of RAM
  - At least 200 MB of hard disk space for binary installation, an additional 600-700 MB for source installation if VC6.0 is not already installed
  - gcc for Unix source installation, VC6.0 for Windows source installation
- Please send support request to [cinema-support@cs.columbia.edu](mailto:cinema-support@cs.columbia.edu).

# パケット通信技術でのリアルタイム性と品質確保の難しさ

```
GoToMeeting Audio Statistics
MODE: VoIP
VE: P M , capture s/w/e 0/0/0, play w/e 0/0, drop 1
VOL: in 0%, out 100%
SPECH: in -24dBm0, out 0dBm0
NOISE: in -81dBm0, out 0dBm0
AGC: gain 0dB, maxLv 255, clip 0ms, convergeLevel 0sec, convergeMic 0sec
ECHO: ERL 0, ERLE 0, RERL -100, A_NLP -100
RES_EC:corr 0%, conf 0%, likelihood 0%, delay median 0ms, std 0
AQE: 80%
CODEC: OPUS (33Kbps, 20ms)
DELAY: est 85ms play 42ms
RTP: avg 2ms, max 24ms, dro 0
RTCP: rtt 0ms
RTCP: frc 0%, cum 42, ext 92984
JB: cur 35ms, tar 20ms
JB: los 0.0%, dis 0.0%
JB: exp 0.0%, pre 0.0%, acc 0.0% AuDrpMs 0, AuDrp 0

FLAGS: AdvNsSupp 1, ReTx 0
PIPE: eSpeakerRole, connected using UDP S R

TRAN: RTP
RTP: in 0Kbps, 92987pkts, 7488KB, dro 0pkts
RTP: out 0Kbps, 0pkts, 0KB, dro 0pkts
RTP: in jit 0ms, los 0.0%/0.0%
RTP: out jit 0ms, los 0.0%/0.0%
RTCP: in 0pkts, 0KB, dro 0pkts
RTCP: out 379pkts, 0KB, dro 0pkts

Vincent [54]
SYS: 9% (3%/6%), proc 0%, dpc 2%, speed 95%
PROC: ca 1%, vc 0% (unk 0%), ui 0% (unk 0%), sum 1%
VOIP: 7%/ 7%/ 7% -> 7% ( 7%/ 0%), 0%, audiodg 0%
SCRE: 6%/12%/25% -> 25% (25%/ 0%), est 0%, up 0%, down 0%
VIDO: 0%/ 0%/ 0% -> 0% ( 0%/ 0%), est 0%, enc 0%, dec 0%, cap 0%, ren 0% mms -1%

BW: up req min 0Kbps, dem 0Kbps, grant sho 0Kbps, lon 0Kbps
BW: dn req min 0Kbps, dem 0Kbps, grant sho 0Kbps, lon 0Kbps
BW: curr: rtt 177ms, ud 88ms, u1 0%, dd 87ms, d1 0%
BW: best: rtt 173ms, ud 86ms, u1 0%, dd 86ms, d1 0%

Video Session: eConferenceIdle
```

## RTPQOS

### NAME

rtpqos - QoS measuring tool for RTP packets

### SYNOPSIS

rtpqos [*options*] remote\_host/local\_port[/remote\_port]

### DESCRIPTION

This is a tool to measure QoS parameters, delay, round trip time, diff, jitter and loss, for RTP packets.

The rtpqos will generate packets with a specific byte pattern which sounds like a periodic beep. The byte pattern will be used as a signature for matching packets later. After sending each packet, it stores the signature of the packet in a table, along with sent timing information.

Then rtpqos listens to packets on a given network address. It determines which original packet received belongs to sent packets by using the embedded byte pattern in the payload.

Calculate and updates delay, jitter and loss statistics to a file.

Consequently, following parameters are created and obtained from the tool, in addition to the information contained in the original RTP headers.

1. From RTP packets:
  1. total round trip delay  
includes processing time at remote.
  2. interval\_from\_prev\_packet
  3. diff  
This is a fraction of deviance from the ideal arrival time.
  4. jitter  
Uses normal 1/16 noise reduction ratio of the diff.
  5. loss  
It is incremented whenever the embedded id jumps.
2. From RTCP packets:
  1. network round trip time  
This is computed from RTCP\_RR\_arrival\_time - DLSR - LSR.  
Contrary to the "total round trip time", it measures round trip time on the network without including processing time at remote.



# Linux kernel のリアルタイム性に苦しむ

- 当時の Linux kernel はタイマーが 10ms の解像度しかなく RTP パケットの 20ms 毎の送信が困難だった。
- Henning 先生より The University of Kansas のリアルタイム性を上げるプロジェクトを参考に自分たちで kernel を触ることに
- 1995年に友人が Linux と FreeBSD を私の PC にインストールして以来、kernel は触れるようになりたかった。

**From:** Henning Schulzrinne <hgs@cs.columbia.edu> **To:** Wenyu Jiang <wenyu@cs.columbia.edu> **Cc:** Akira  
**Subject:** Re: Weekly Report **Date:** Mon, 19 Aug 2002 10:49:21 -0400

<x-flowed> <http://www.ittc.ku.edu/kurt/> also looks promising and newer.

Wenyu Jiang wrote:

> Hi, Henning

>

> Yes, we did try at one time running newudpl on Linux, the result was

> similar to Solaris 2.8, with the 10ms effect. We used njt, which has

> Linux 2.4 kernel (Red-hat 7.2).

>

> <http://hegel.ittc.ukans.edu/projects/posix/time.html>

>

> mentions an extension to Linux with the i

> am not sure if it's installed on clic machine:

> relevant), or it works on more recent kerne

>

> We will check whether the Solaris timer f

>

> Thank you.

>

> Wenyu

>

> On Sun, 18 Aug 2002, Henning Schulzrinne

>

>

>> Another question is whether Linux is any

>> reference I sent earlier did not seem to m

>>

> </x-flowed>

## KURT-Linux News

Last modified: Mon Oct 6 13:42:19 CDT 2003

KURT 2.4.18 Version 2 has been added to the [downloads section](#).

A patch that includes iPAQ-specific modifications has been added to the [downloads section](#).

A patch to Linux kernel 2.4.18 is now available. This is the latest release of KURT-Linux, including the UTIME microsecond timing extensions, the KURT real-time scheduling extensions, and the DSKI information gathering facility.

We have also made available a [draft of the KURT-Linux User Manual](#).

We are in the midst of a web-page overhaul. To view the old, soon to be replaced page, [go here](#).

## Mailing List

Subscribe to the KURT-Linux mailing list by sending an email to [majordomo@ittc.ku.edu](mailto:majordomo@ittc.ku.edu) with the body

subscribe linux-kurt <your-email-address>

[KURT @ ITTC](#)



# インターネット回線の混雑状況を再現

## NEWUDPL

### NAME

newudpl - Network Emulator With UDP Link

### SYNOPSIS

```
newudpl [-v|vv] [-p [recv_port]:[send_port]]  
        [-i source_host[:[/][port]*]] [-o dest_host[:[/][port]]]  
        [-s link_speed] [-d delay] [-e Ethernet_speed] [-q queue_buf_size]  
        [-B|L|C|U|O] error_rate]
```

### DESCRIPTION

This is a tool to create various condition of packet switching behavior artificially for UDP packets.

The emulator receives UDP packets on a designated port and then delays them, drops some randomly, corrupts the content, swaps the order of sending and emulates a finite-bandwidth link by queuing packets. It should be useful to examine an efficiency of network protocols or codecs.

### INSTALLATION

Source codes are available from:  
<http://www.columbia.edu/~at541/src>

#### Unix/Unix like system

To build, unpack the tar file, then type:

```
./configure  
make
```

#### Windows98/NT

Microsoft VC++ 6

Open newudpl.dsw under msc\ and just press F7.

Borland C++ builder

Open newudpl\_bcb.bpr under bcb\ and just press F9.

I have only tested on SunOS 5.7, 5.8 system and Linux but it should compile on other Posix-compliant platforms.

下記のパラメーターを設定可能

- パケットロス
- 遅延
- パケットの順不同到着
- 各パケットの到着時間の揺れ(jitter)

次ページの rtpqos を使って音声と動画の品質を最適化

授業の教材になりうれしかった

# 音声・動画データの品質計測

## RTPQOS

### NAME

rtpqos - QoS measuring tool for RTP packets

### SYNOPSIS

rtpqos [[options](#)] remote\_host/local\_port[/remote\_port]

### DESCRIPTION

This is a tool to measure QoS parameters, delay, round trip time, diff, jitter and loss, for RTP packets.

The rtpqos will generate packets with a specific byte pattern which sounds like a periodic beep. The byte pattern will be used as a signature for matching packets later. After sending each packet, it stores the signature of the packet in a table, along with sent timing information.

Then rtpqos listens to packets on a given network address. It determines which original packet received belongs to sent packets by using the embedded byte pattern in the payload.

Calculate and updates delay, jitter and loss statistics to a file.

Consequently, following parameters are created and obtained from the tool, in addition to the information contained in the original RTP headers.

#### 1. From RTP packets:

1. total round trip delay  
includes processing time at remote.
2. interval\_from\_prev\_packet
3. diff

This is a fraction of deviance from the ideal arrival time.

RTP と RTCP パケットから  
下記のパラメーターを計測  
可能

- 往復の遅延時間
- 理想到着時間からの実際の到着時間のずれ(diff)
- 各パケットの到着時間の揺れ(jitter)
- パケットロス

# 乱数生成関数のランダム性で苦勞する

```
21 ↓
22 /** random function.↓
23  * returns uniform distributed random num
24  * but it is slower then ANSI rand() func
25  * From↓
26  * 'Numerical Recipes in C'↓
27  * The Press Syndicate of the University
28 float ran1(long *idum);↓
29 ↓
30 /** binomial random function.↓
31  * returns random number according to binomial-distribution.↓
32  * From↓
33  * 'Numerical Recipes in C'↓
34  * The Press Syndicate of the
35 int bnldev(float pp, int n, lo
36 ↓
37 ↓
38 /** Gilbert random function.↓
39  * returns 1 for packet loss,
40 int gilbRand(float pc, float p
41 ↓
```

自分で作った乱数  
生成ライブラリ

```
/*
 * Generate a random 32-bit quantity.
 */
#include <sys/types.h>
#include <sys/time.h>
#include <unistd.h>
#include <stdio.h>
#include <time.h>
#include <sys/utsname.h>
#include "global.h"
#include "md5.h"

#define MD_CTX MD5_CTX
#define MDInit MD5Init
#define MDUpdate MD5Update
#define MDFinal MD5Final

static u_long md_32(char *string, int length)
{
    MD_CTX context;
    union {
        char    c[16];
        u_long  x[4];
    } digest;
    u_long r;
    int i;

    MDInit (&context);
```

IETF RTP プロトコル  
RFC 乱数生成のコード  
例

Schulzrinne, et al.  
RFC 3550

Standards Track  
RTP

[Page 85]  
July 2003

- o getdomainname(),
- o getwd(), or
- o getrusage().

IETF RTP プロトコル  
RFC での乱数生成ガ  
イド

" video or audio sa  
rs, but care must be taken to avoid using a turned-off  
phone or blinded camera as a source [17].

f this or a similar routine is recommended to generate the  
al seed for the random number generator producing the RTCP  
d (as shown in Appendix A.7), to generate the initial values for  
equence number and timestamp, and to generate SSRC values.  
this routine is likely to be CPU-intensive, its direct use to  
ate RTCP periods is inappropriate because predictability is not  
sue. Note that this routine produces the same result on  
ted calls until the value of the system clock changes unless  
rent values are supplied for the type argument.

MD5 のセキュリティバグ発見につながる

一様分布する乱数生成関数は遅い  
コメントアウトしたら EU から疑われた 17

# md5 実装にセキュリティバグを見つける

```
112 operation, processing another message block, and updating the
113 context.
114 */
115 void MD5Update (context, input, inputLen)
116 MD5_CTX *context;          /* context */
117 unsigned char *input;       /* input block */
118 unsigned int inputLen;      /* length of input block */
119 {
120     unsigned int i, index, partLen;
121     /* Compute number of bytes mod 64 */
122     index = (unsigned int)((context->count[0] >> 3) &
123     /* Update number of bits */
124     if ((context->count[0] += ((UINT4)inputLen << 3))
125         context->count[1]++;
126     context->count[1] += ((UINT4)inputLen >> 29);
127     partLen = 64 - index;
128     /* Transform as many times as possible. */
129     if (inputLen >= partLen) {
130         memcpy
131         ((POINTER)&context->buffer[index], (POINTER)input,
132         MD5Transform (context->state, context->buffer);
133     }
134     for (i = partLen; i + 63 < inputLen; i += 64)
135         MD5Transform (context->state, &input[i]);
136     index = 0;
137     }
138     else
139     {
140         i = 0;
141         /* Buffer remaining input */
142         /* fixed by Akira Tsukamoto 04/04/2002 */
143         if (i >= inputLen)
144             return;
145         /* end fix */
146         memcpy
147         ((POINTER)&context->buffer[index], (POINTER)&input[i],
148         inputLen-i);
149     }
150 }
```

今でいう CVE に相当

# TEEP Protocol の技術開発の傍ら

- AIST で rtptools のメンテナーをしたかった
  - <https://github.com/irtlab/rtptools>

## RFC3550

### RTP:

## A Transport Protocol for Real-Time Applications

☰ README.md

## RTP Tools

RTP Tools is a set of small applications that can be used for processing RTP data. Refer to the individual manpages for details.

- **rtplay** play back RTP sessions recorded by rtpdump
- **rtpsend** generate RTP packets from textual description, generated by hand or rtpdump
- **rtpdump** parse and print RTP packets, generating output files suitable for rtplay and rtpsend
- **rtpttrans** RTP translator between unicast and multicast networks
- **multidump** Start multiple rtpdumps simultaneously.
- **multiplay** Start multiple rtplays simultaneously.

## Installation

RTP tools should compile and run on any POSIX compatible system, as well as on Windows. Some operating systems also provide a prebuilt package of RTP tools.

On UNIX, the usual `./configure && make` should work. Read on for the details.

### configure

Run `./configure` to configure the build for your system. This will produce three files:

- `config.h` containing the `#include` and `HAVE_` lines
- `config.log` containing the details of autodetection
- `Makefile.local` which defines `CC`, `PREFIX` and the like

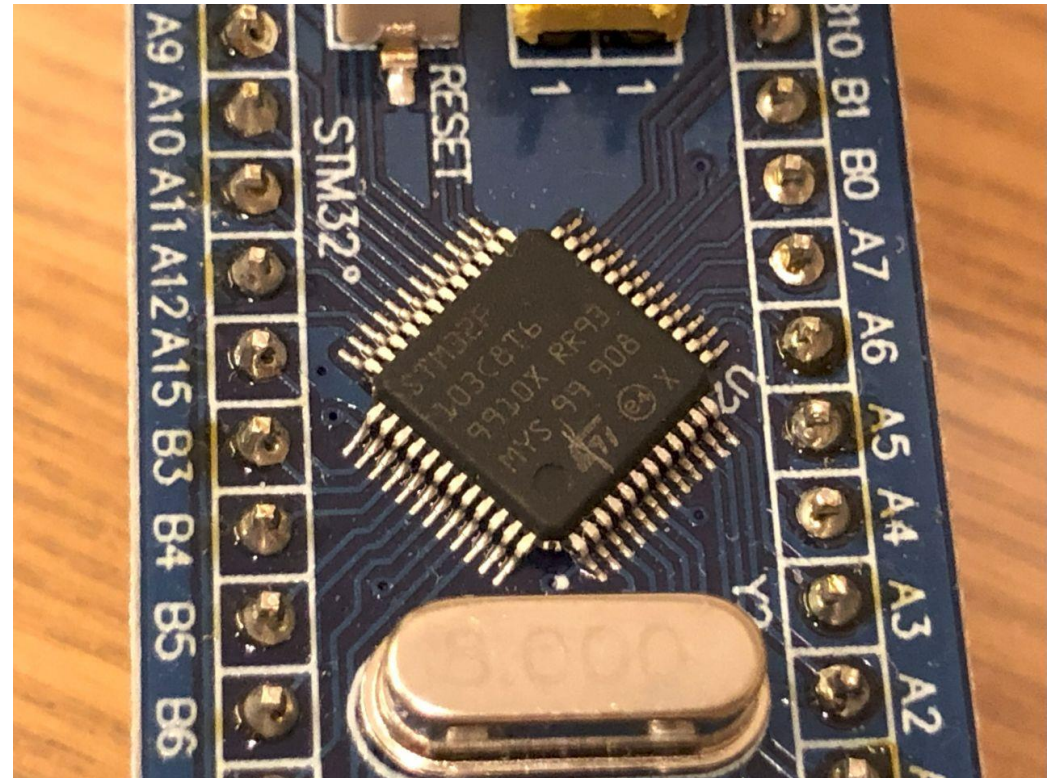
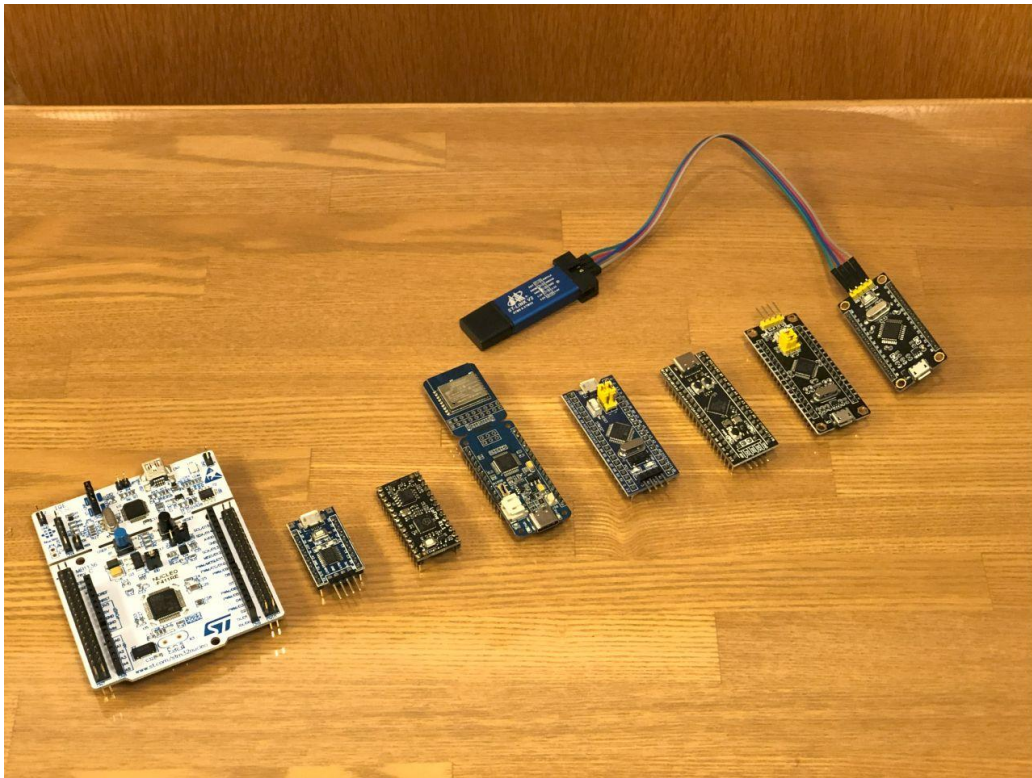
Read the standard output and `Makefile.local`. If these look different from what you expected, read



# その他の活動 (1/2)

STM32 Blue Pill, Black Pill を Arduino IDE を使わずに  
コマンドラインで開発できる環境を提供することを目的

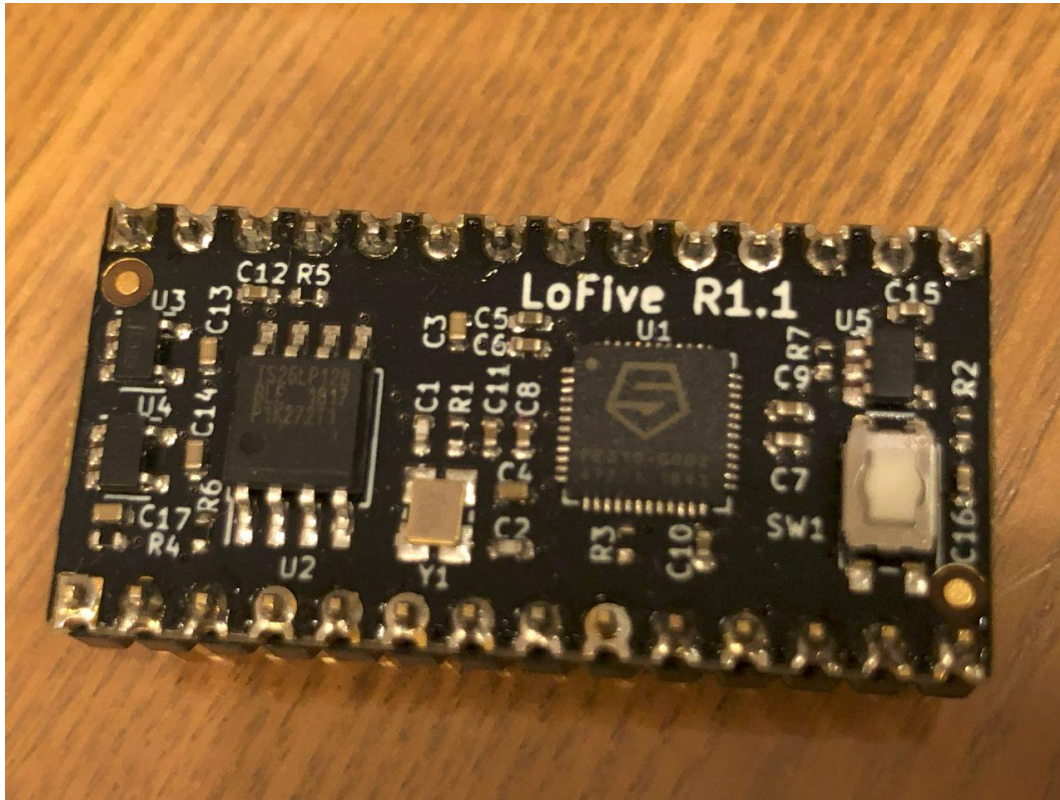
<https://github.com/mcd500/arduino-commandliners>





## その他の活動 (2/2)

- 将来的に他の STM32 や RISC-V チップもサポートしたい



SiFive FE310



GD32VF103

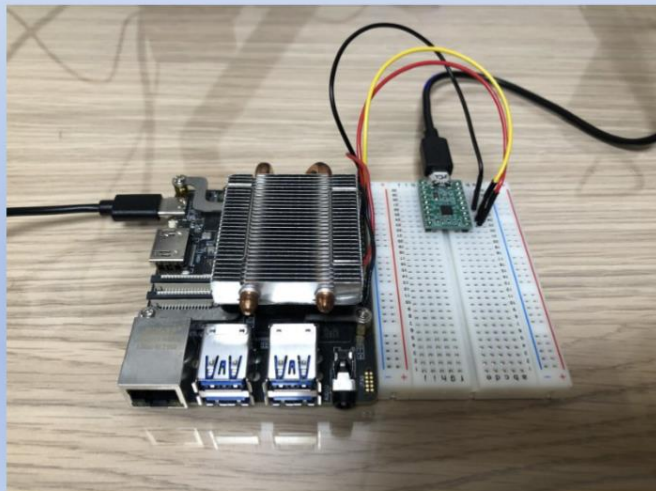
# RISC-V の高速化 (1/3)

- [https://static.sched.com/hosted\\_files/riscvsummit2021/fe/2021-12-05-RISC-V-Summit-BOF-ucopy-AkiraTsukamoto-4.pdf](https://static.sched.com/hosted_files/riscvsummit2021/fe/2021-12-05-RISC-V-Summit-BOF-ucopy-AkiraTsukamoto-4.pdf)
- CPU 使用率が 34.69% (memcpy), 33.84% (copy\_to\_user) 計68.53% と極端に高く、これが原因でネットワーク速度が遅くなっていた。

## What was observed on RISC-V RV64 boards

- Starlight

- SoC: StarFive JH7100
- Core: SiFive U74 (Dual core)



- Workload of Network benchmark

- iperf3 -u -b 1000M --length 6500 -c 192.168.1.112

- On starlight, similar on Unmatched

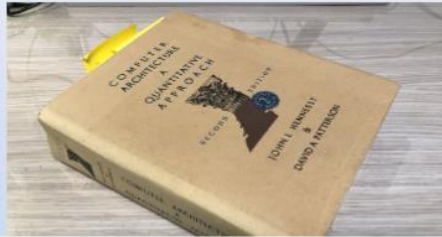
- perf top -Ue task-clock

```
Samples: 35K of event 'task-clock', 4000 Hz, Event count (
Overhead Shared D Symbol
34.69% [kernel] [k] memcpy
33.84% [kernel] [k] __asm_copy_to_user
2.28% [kernel] [k] sifive_l2_flush64_range
1.58% [kernel] [k] dev_gro_receive
1.41% [kernel] [k] skb_gro_receive
1.28% [kernel] [k] memset
0.97% [kernel] [k] _raw_spin_unlock_irqrestore
0.86% [kernel] [k] page_pool_put_page
0.75% [kernel] [k] finish_task_switch.isra.0
0.66% [kernel] [k] __skb_datagram_iter
0.63% [kernel] [k] inet_gro_receive
0.60% [kernel] [k] enh_desc_get_rx_status
0.55% [kernel] [k] get_page_from_freelist
```

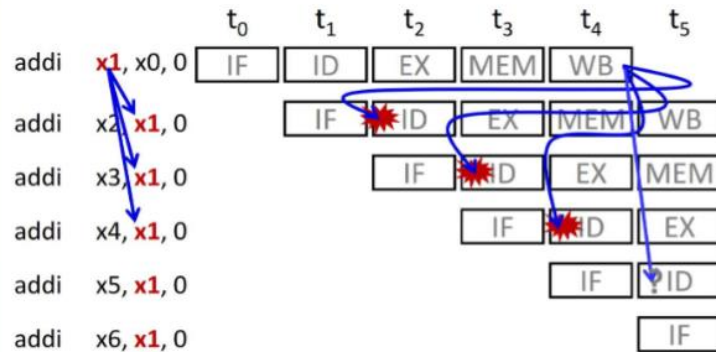


# RISC-V の高速化 (2/3)

“Computer Architecture: A Quantitative Approach”



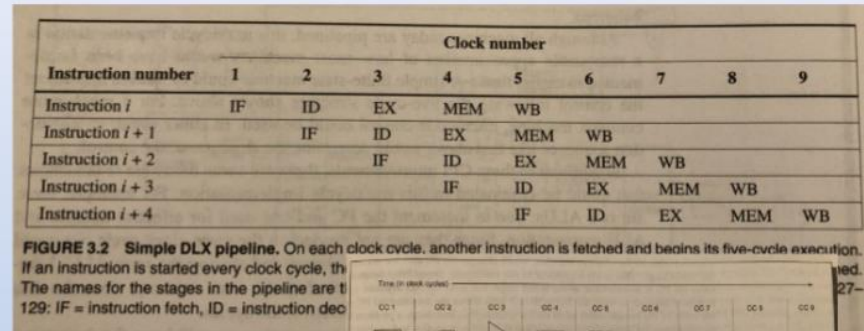
## Dependency and Hazard: e.g. RAW



<https://users.ece.cmu.edu/~jhoe/course/ece447/S10handouts/L08.pdf>



Able to predict U74 would likely to have 3 pipeline stall for RAW from 4 cycle load use  
Good to study on good book!



Describes condition of  
read after write data hazard  
(RAW)

# RISC-V の高速化 (2/3)

ただのコピー関数なのに、ソースのサイズがやたら大きい、RISC系 CPU の特徴

- <https://elixir.bootlin.com/linux/latest/source/arch/riscv/lib/uaccess.S>

## The code, putting all together

The location of the source of copy\_to\_user(), copy\_from\_user().

<https://elixir.bootlin.com/linux/latest/source/arch/riscv/lib/uaccess.S>

```
/*
 * Register allocation for code below:
 * a0 - start of uncopied dst
 * a1 - start of uncopied src
 * a2 - size
 * t0 - end of uncopied dst
 */
add t0, a0, a2

/*
 * Use byte copy only if too small.
 * SZREG holds 4 for RV32 and 8 for RV64
 */
li a3, 9*SZREG
bltu a2, a3, .

/*
 * Copy first byte
 * a0 - start of
 * t1 - start of
 */
addi t1, a0, 5
andi t1, t1, ~
/* dst is already
beq a0, t1, .

1:
/* a5 - one byte for copying data */
fixup lb a5, 0(a1), 10f
addi a1, a1, 1 /* src */
fixup sb a5, 0(a0), 10f
addi a0, a0, 1 /* dst */
bltu a0, t1, 1b /* t1 - start of aligned dst */

.lskip_align_dst:
/*
 * Now dst is aligned.
 * Use shift-copy if src is misaligned.
 * Use word-copy if both src and dst are aligned because
 * can not use shift-copy which do not require shifting
 */
/* a1 - start of src */
andi a3, a1, SZREG-1
bnez a3, .lshift_copy
```

Checking src and dst are  
8 byte aligned address or  
not, if not, copy until dst  
is aligned

```
.lshift_copy:
/*
 * Word copy with shifting.
 * For misaligned copy we still perform aligned word copy, but
 * we need to use the value fetched from the previous iteration and
 * do some shifts.
 * This is safe because reading is less than a word size.
 */
/* a0 - start of aligned dst
 * a1 - start of src
 * a3 - a1 & mask:(SZREG-1)
 * t0 - end of uncopied dst
 * t1 - end of aligned
 */
/* calculating aligned
andi t1, t0, ~(SZREG-1)
/* Converting unalign
andi a1, a1, ~(SZREG-1)

/*
 * Calculate shifts
 * t3 - prev shift
 * t4 - current shift
 */
slli t3, a3, 3 /*
li a5, SZREG*8
sub t4, a5, t3

/* Load the first word
fixup RREG_L a5, 0(a1)

2:
/* Main shifting copy
 * a0 - start of align
 * a1 - start of align
 * t1 - end of align
 */
/* At least one iteration
srl a4, a5, t3
fixup RREG_L a5, SZREG
addi a1, a1, SZREG
slli a2, a2, t4
or a2, a2, a4
fixup RREG_S a2, 0(a0)
addi a0, a0, SZREG
bltu a0, t1, 3b

/* Revert src to original unaligned value */
add a1, a1, a3
```

The src is not aligned,  
read src every 8 byte in  
aligned address but  
shifting data in registry  
to compensate of  
reading unaligned data  
on closest aligned  
address

```
.lword_copy:
/*
 * Both src and dst are aligned, unrolled word copy
 */
/*
 * a0 - start of aligned dst
 * a1 - start of aligned src
 * t0 - end of aligned dst
 */
addi t0, t0, -(8*SZREG) /* revert to original value */

2:
fixup REG_L a4, 0(a1)
fixup REG_L a5, SZREG(a1)
fixup REG_L a6, 2*SZREG(a1)
fixup REG_L a7, 3*SZREG(a1)
fixup REG_L t1, 4*SZREG(a1)
fixup REG_L t2, 5*SZREG(a1)
fixup REG_L t3, 6*SZREG(a1)
fixup REG_L t4, 7*SZREG(a1)
fixup REG_S a4, 0(a0)
fixup REG_S a5, SZREG(a0)
fixup REG_S a6, 2*SZREG(a0), 10f
fixup REG_S a7, 3*SZREG(a0), 10f
fixup REG_S t1, 4*SZREG(a0), 10f
fixup REG_S t2, 5*SZREG(a0), 10f
fixup REG_S t3, 6*SZREG(a0), 10f
fixup REG_S t4, 7*SZREG(a0), 10f
addi a0, a0, 8*SZREG
addi a1, a1, 8*SZREG
bltu a0, t0, 2b

addi t0, t0, 8*SZREG /* revert to original value */
j .lbyte_copy_tail
```

Both src and dst are  
aligned, perform  
unrolled copy with every  
8 byte in aligned address

# RISC-V で苦労したところ

- 業界のニーズにミスマッチが多い。
- RISC-V はオープンソース？
  - 技術的には RISC-V はオープンソースでは無いです。
  - RISC-V の命令セット (Instruction Set Architecture, ISA) が公開されている、オープン規格であるのが真相。
    - Krste Asanovic 先生は Open Specification と呼ぶ  
<https://www.youtube.com/watch?v=RrVRMFjYti0&t=696s>
    - David Patterson 先生は Open Standard と呼ぶ  
<https://riscv.org/blog/2023/03/top-ten-fallacies-about-risc-v/>
- オープンソースは何がメリット？
  - 盛り上がったプロジェクトは開発速度が速い、それに尽きる。
  - GAFAM の体力をもってしてもクローズドな開発では追い付かない。
  - 半導体企業などプラットフォームをとりたい企業が有効活用。
  - もちろん各企業のビジネスでの取り組みであるため、すべてをオープンソースにはしない。
- 日本の半導体業界
  - 海外の技術や製品を買ってくるだけの話になりがちで、、、

# Appendix



# 中学校 1 年生の時にマイコン雑誌に出会う





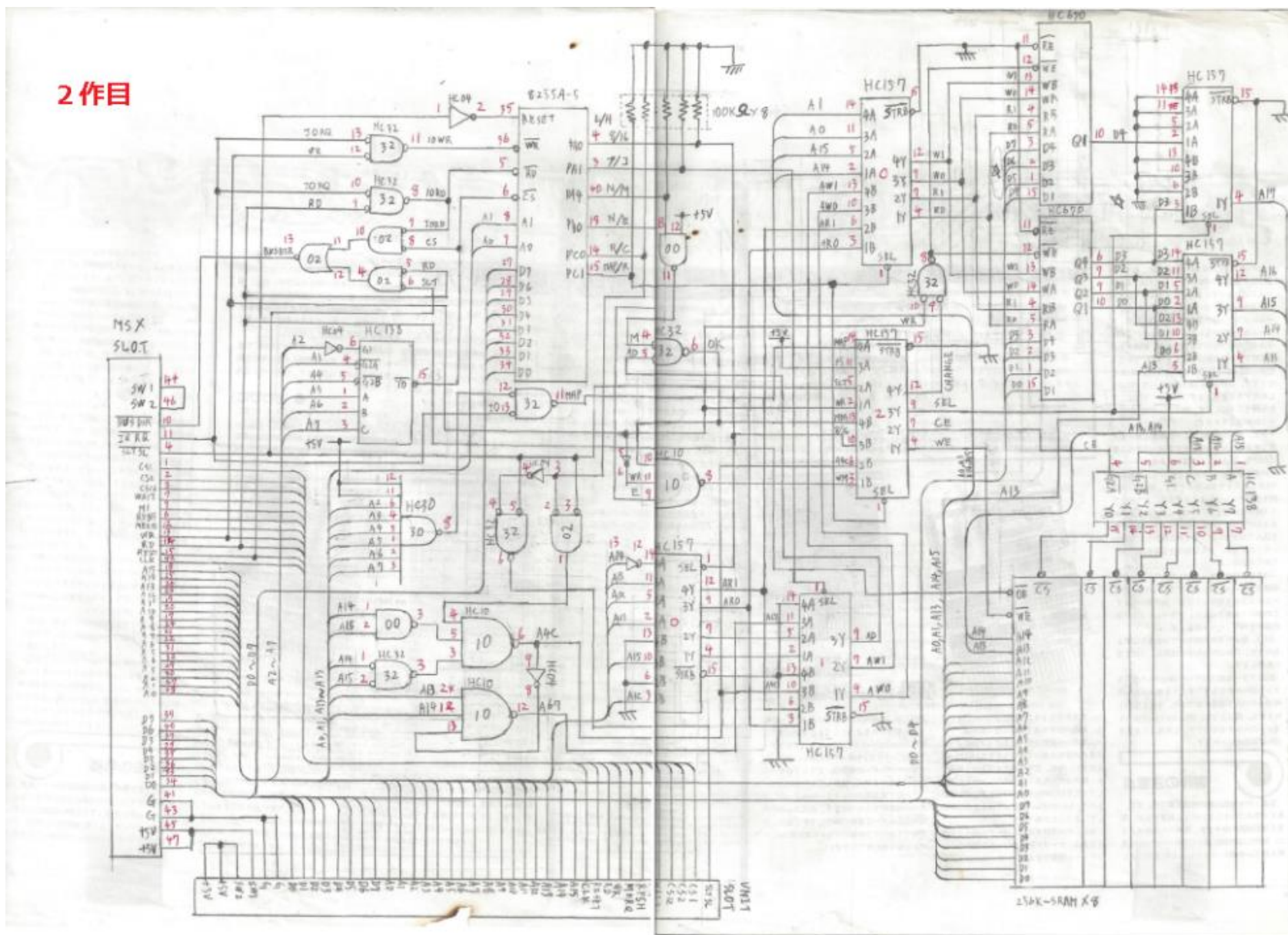




## 2か月の試行錯誤の後に動いた。写真は2作目



## 初代と2作目の機能は同じ、最適化進む





# アセンブラ言語をハンドアセンブルで実行した



Date		No. 1	
SAVE			
F3		DI	
2A, 00, D5		LD	HL, (00500)
3A, 03, D5		LD	A, (00503)
55		PUSH	HL
CD, 24, 00		CALL	00 24
E1		POP	HL
7E		LB	A, (HL)
4F		LD	C, A
86, 0A		LD	B, 0Ah
ED, A1	LOOP, 1	CPI	
20, 1C		JR	NZ, ORIGINAL
18, F8		JR	LOOP, 1
FE, EA		CP	EAh
2E, 0A		JR	Z, LOAD
FE, 08		CP	DDh
2E, 0C		JR	Z, C, BLOAD
FE, 03		CP	D3h
2E, 0B		JR	Z, C, BLOAD
18, 0C		JR	ORIGINAL
0E, 02	LOAD, 1	LD	C, 02h
0A, 0F		LD	B, 0Fh
18, 0A		JR	INPUT
0E, 01	C, BLOAD, 1	LD	C, 01h
0A, 0F		LD	B, 0Fh
18, 0A	ORIGINAL, 1	JR	INPUT
0E, 00		LD	C, 00h
3E, 02		LD	A, 02h
2A, 00, D5	INPUT	LB	HL, (D500)
C5	LOOP, 2	PUSH	3C
55		PUSH	HL
CD, EA, 00		CALL	DEEA
E1		POP	HL
C5		POP	BC

① D500 ← ST. AD,  
D502 ← EN. AD,  
D503 ← W. XAM, SL, NS,  
② D504 ← CR. (0h) or ER. (Xh)  
③ AF, BC, DE, HL,

LOOP

TOTAL



# 九州歯科大学

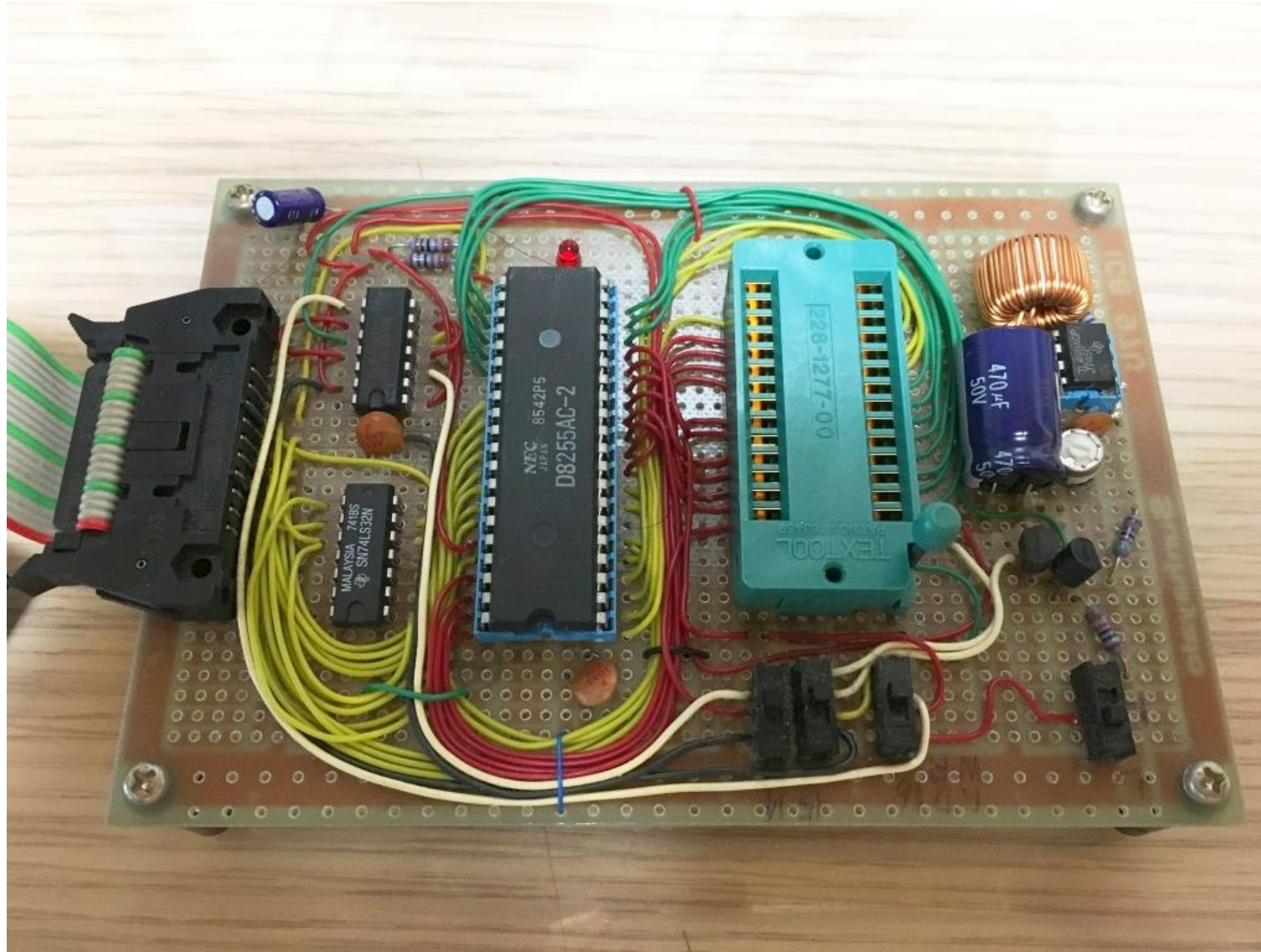


第39回 九州歯科大学体育祭 学部2年

平成3年10月24日



# 九州歯科大学に在学中もデジタル技術を。

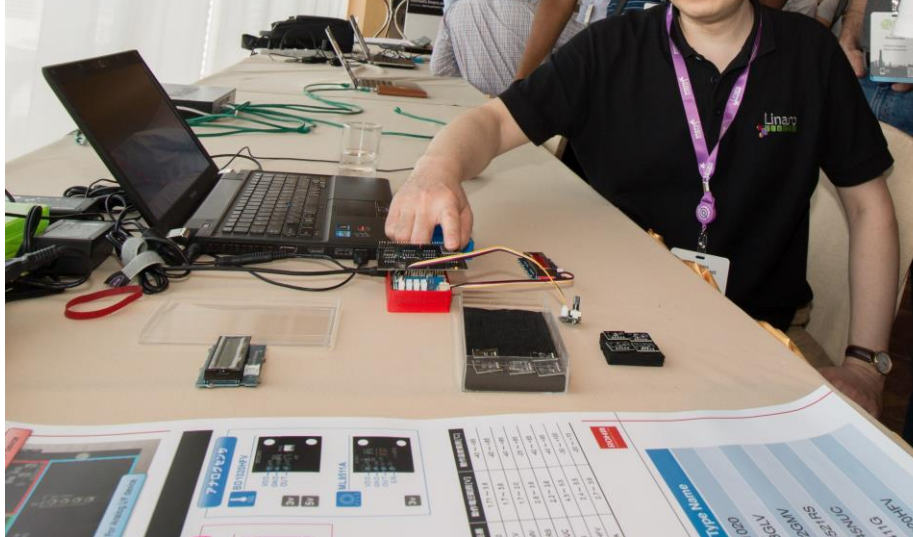


# 歯科医師していた時の自宅の机(笑)





# Linaro で。SCEの時のお知り合い多く





# ARM チップに SPI 接続のセンサー制御

```
mraa:spi16: rx=0000
Success to read addr fc data 00
mraa:spi16: tx=001b
mraa:spi16: rx=0000
Success to write addr 1b data 00
mraa:spi16: tx=00fb
mraa:spi16: rx=0000
Success to read addr fb data 00
mraa:spi16: tx=00f5
mraa:spi16: rx=0000
Success to read addr f5 data 00
main(); whoami = 00
Success to read buf addr fb data 00
Accelerometer: AX=0.000000 AY=0.312500 AZ=0.171387
Gyrometer:      GX=-27.412214 GY=-2.091603 GZ=21.442749
```



# 自宅で作業時、家が散らかるので怒られます、、、

